# Improving the Sampling Strategy for Point-to-Point Line-Of-Sight Modelling in Urban Environments

# Improving the Sampling Strategy for Point-to-Point Line-Of-Sight Modelling in Urban Environments

*Phil Bartie and William Mackaness*
*email: phil.bartie@stir.ac.uk*

**Abstract:** Visibility modelling calculates what an observer could theoretically see in the surrounding region based on a digital model of the landscape. In some cases it is not necessary, nor desirable, to compute the visibility of an entire region (i.e. a viewshed), but instead it is sufficient and more efficient to calculate the visibility from point-to-point, or from a point to a small set of points, such as computing the intervisibility of predators and prey in an agent based simulation. This paper explores how different line-of-sight (LoS) sample ordering strategies increases the number of early target rejections, where the target is considered to be obscured from view, thereby improving the computational efficiency of the LoS algorithm. This is of particular importance in dynamic environments where the locations of the observers, targets and other surface objects are being frequently updated. Trials were conducted in three UK cities, demonstrating a robust five-fold increase in performance for two strategies (hop, divide and conquer). The paper concludes that sample ordering methods do impact overall efficiency, and that approaches which disperse samples along the LoS perform better in urban regions than incremental scan methods. The divide and conquer method minimises elevation interception queries, making it suitable when elevation models are held on disk rather than in memory, while the hopping strategy was equally fast, algorithmically simpler, with minimal overhead for visible target cases.

***Keywords:*** *visibility analysis; LBS; urban modelling; line of sight; sample ordering*

## 1 Introduction

Visibility modelling is used to calculate what is theoretically visible from a specified location, with consideration given to the observer's height and surrounding topography. Increasingly surface objects (e.g. buildings, vegetation) are included in surface models

derived from Light Detection and Ranging (LiDAR) point clouds, expanding the range of uses of visibility analysis in urban regions. Viewshed analysis calculates the regions visible from a specific location, requiring every cell in a raster surface to be accessed making it computationally expensive. However it is not always necessary to calculate the visibility from an observer to a region, but instead the visibility between defined point locations. For example computing the intervisibility between vehicles in an urban transport simulation, or animals in a predator-prey agent based model (ABM), requires visibility to be modelled from each observer's viewpoint. In such cases computing the viewshed (i.e. region visible) for each observer would be very computationally expensive and highly inefficient as much of the calculation time would be spent computing the visibility of cells not relevant to the result. This is particularly relevant in dynamic multi-observer simulations where the locations of agents (e.g. cars, people) are frequently updated. In these instances point-to-point ray casting offers a more suitable solution, by determining if an unbroken line-of-sight (LoS) exists between set members.

This research assesses the impact that the sample order has on performance in point-to-point LoS calculations for urban regions, where only a Boolean target visibility result is required (i.e. target is visible or not visible). The paper begins with a review of visibility modelling, followed by a short introduction to the LoS model implementation, before exploring a variety of LoS sampling strategies. The sample order strategies are variations of the order in which points along the LoS are tested to determine if the target is visible or not. An ideal sampling strategy would be one that consistently resulted in early rejection of obscured targets (i.e. it avoids scanning all intermediate elevation values between object and target). Five different sampling orders were assessed across three UK cities, concluding that fivefold performance gains were possible where a divide and conquer or hopping method were used. The hopping method included a parameter for hop size, and trials showed that hop sizes of 20 to 30 metres were optimal in urban regions. An explanation for this was sought by constructing synthetic city elevation models of varying road width, revealing a strong positive correlation between

hop size and road width. The paper concludes with comments about how the LoS performance improvement may be used, with suggestions for follow up research.

## 2 Background

Visibility modelling is included in the majority of Geographic Information Systems (De Smith *et al.* 2007), and has become one of the most commonly used analysis tools (Davidson *et al.* 1993). It is used for a range of research including landscape planning (Fisher 1996), locating the most scenic or most hidden routes (Stucky 1998), siting radio masts and wind turbines (De Floriani *et al.* 1994a), modelling spatial openness in built environments (Fisher-Gewirtzman and Wagner 2003), and in military exercises as a weapon surrogate (Baer *et al.* 2005).

Isovists (Tandy 1967, Benedikt 1979, Turner *et al.* 2001) have tended to be used in modelling urban visibility where a map of building footprints is available but for which there is no height information. In these cases the heights of the buildings are considered to be infinite, and the limits of visibility are determined by a building's walls. A better approximation of visibility is possible when a Digital Surface Model (DSM) is available, typically collected using LiDAR, offering a 2.5D dataset that includes the height of surface objects such as buildings and vegetation. In these cases a viewshed (Tandy 1967, Lynch 1976) may be calculated which shows the regions visible from a specified observation location.

The computational efficiency of isovist and viewshed models has received much attention (De Floriani *et al.* 2000, Rana and Morley 2002, Rana 2003, Ying *et al.* 2006). If every terrain cell in a line-of-sight path is considered between an observer and target it is referred to as the 'golden case' (Rana and Morley 2002). This approach can be computationally expensive; techniques have therefore been developed to reduce the number of calculations by considering only visually important cells. For example a Triangulated Irregular Network (TIN) (De Floriani and Magillo 1994) represents the

terrain as triangles and can be used to reduce surface noise by only depicting large elevation changes. This and other terrain filtering techniques, can be used to reduce the number of observer-target pairs considered in viewshed generation (Rana and Morley 2002) but are more suited to rural landscapes than the densely varied urban landscape.

The efficiency of viewshed algorithms may also be improved (Seixas *et al.* 1999), including strategies such as sweeping LoSs to a set of perimeter cells a fixed distance from the observer (Franklin and Ray 1994), replacing sightlines with reference planes (Wang *et al.* 2000), and using voxels (Carver and Washtel 2012). There are also benefits from maintaining partial blocking information using a balanced binary search tree (Van Kreveld 1996), or lists (Andrade *et al.* 2011), to reduce duplication in the computation of visibility for multiple LoSs passing over a cell.

Furthermore processing times may be reduced by parallelisation across multiple cores (De Floriani et al. 1994b), across distributed systems (Mills et al. 1992), or on a Graphics Processing Unit (GPU) (Xia et al. 2010, Gao et al. 2011). The reduction in wall-clock time is not due to greater efficiency but from the computation being divided into tasks that run concurrently on multiple processing cores.

Historically visibility modelling research has focussed on rural settings using digital elevation models of the bare earth topography, but the situation has changed with the availability of LiDAR datasets that also capture surface features (e.g. buildings). The morphology of urban regions is quite different to rural regions, with more rapid elevation changes between buildings and streets, forming more densely packed ridges and valleys (Gal and Doytsher 2012). While the best views in rural areas are from the higher elevations (hilltops), in urban space the observer is usually in the equivalent of a 'valley', between the buildings, where dramatic changes in the visibility of features and landmarks can fall in and out of view in a matter of a few strides. As a result there is a need to revisit visibility modelling algorithms for a range of new uses in these environments. Consider for example the requirement of a location based game running on a smartphone, which can model user intervisibility. For such an application each user would provide a location (point) and require a list of which users should be visible,

without the cost (limited CPU and battery life) of determining all surrounding visible regions. Other applications of point-to-point visibility modelling include modelling the view to junctions, traffic and signals, for vehicle navigation systems (Bartie and Kumler 2010, Tarel *et al.* 2012), GNSS shadow matching (Groves 2011), label removal for obscured features in Augmented Reality applications (e.g. Nokia City Lens), and simulations where more lifelike interactions are modelled by including visibility (e.g. predator - prey ABM, transport simulations). In these cases each observer and target can be adequately represented by a point location on the map surface, and visibility calculated using a LoS between those locations. This paper focuses on that aspect of visibility modelling, to improve the efficiency of point-to-point LoS modelling for urban environments in anticipation of their growing use within highly dynamic applications.

# 3 Line of Sight Calculation

LoS algorithms compare the vertical angle created from an observer to a specified target at another location, against the vertical angles from the observer to all cells in between (Fisher 1993). To reduce the computational cost, only the ratios need to be compared rather than actual angles. If any intermediate cell creates a viewing ratio greater than that of the observer to target ratio, then the target is considered to be obscured (Figure 1). The assumption is that the target is considered to be visible until proven otherwise, and that the ratio from observer to target is the first calculation to which all other ratios are compared. As soon as an intermediate viewing ratio is calculated above that of the target, then the search may be aborted as the target has been shown to be obscured from the observer (for example in Figure 1, $0.75 > 0.167$ indicating an obscured target).

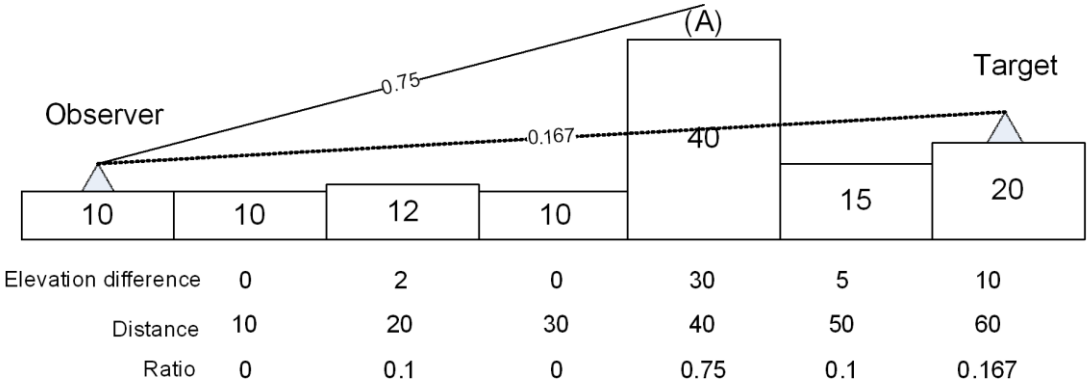



*Figure 1: A line-of-sight approach calculating view ratios from observer to target. The ratio from observer to target is lower than to cell (A) therefore the target is obscured.*

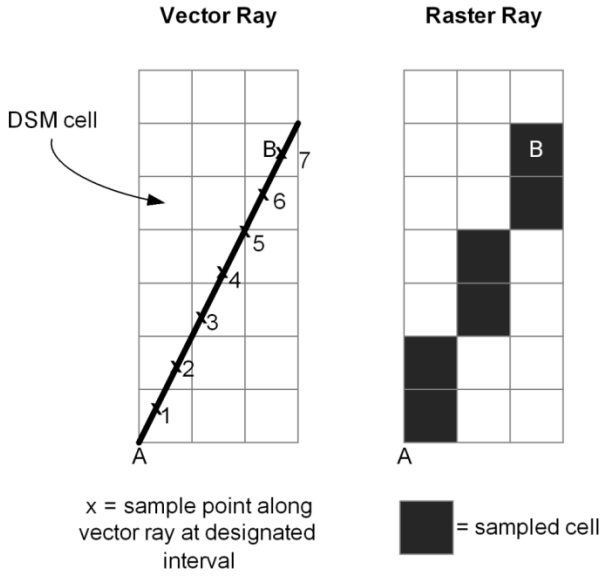
If every terrain cell in a line-of-sight path is considered between an observer and target it is referred to as the 'golden case' (Rana and Morley 2002), but for a Boolean point-to-point visibility result these intermediate values are not required. A simple sampling strategy is to check the ratio for each cell incrementally from the observer to the target and determine if the target cell is obscured. If it is not blocked then the next intermediate cell is tested, until either all cells along the line-of-sight have been checked and the target is considered visible, or the target is calculated to be below the current view ratio and therefore obscured from view, resulting in early scan termination. This research seeks to find if the sample ordering strategy yields performance gains for point-to-point LoS calculations in urban environments.

It is important to acknowledge the impact of data structure and type on the implementation. Elevation data may be stored as a Digital Elevation Model (DEM) or a Triangulated Irregular Network (TIN) (Lee 1991); each model has associated benefits and drawbacks (Kumler 1994). For this urban study a high resolution (1 metre) DSM, based on a LiDAR dataset, was found to be more suitable than using a TIN equivalent. To maintain the vertical resolution the TIN equivalent become extremely large, and despite using spatial indexes the ray-surface intercept performance was found to be significantly inferior. This was probably a result of the complexity of the urban

morphology which can be more easily represented in a cell based data format where indexes are implicit, and ray-surface intercepts may be retrieved more rapidly.

# 4 Line of Sight Sampling

There are a number of ways to sample the values from DSM cells between an observer and target. These include using a vector line which is sampled at given intervals along its length (Figure 2a – vector ray), and a raster approach such as the Bresenham's line algorithm, which uses integer addition to determine sample cells in order along a path from an observer to a designated target (Figure 2b – raster ray).




*(a)* *(b)*

*Figure 2: Cell sampling approaches based on (a)vector and (b) raster lines*

For this research the vector approach was adopted due to the ease with which sample locations can be re-ordered. Samples were located by projecting a point from the observer (Figure 2a - point A) a given distance, determined by the sample method used, at a specified angle towards the target (Figure 2a - point B). The sample ordering approaches used are illustrated in Figure 3. These were the normal forward incremental ordering (A) and reverse ordering (B), to test if obstructions near the observer or target

would result in early-target rejection. A first/last (C) sampling strategy was also included to give equal weighting to cells near the observer and cells near the target. For a more even priority across the length of the LoS a divide and conquer sample approach was used (D) which recursively divided the LoS by half, and a hopping strategy (E) whereby samples were taken at intervals (e.g. every 2m) along the LoS from the observer to the target. In cases where the target was visible each method would result in all intermediate sample locations being scanned.

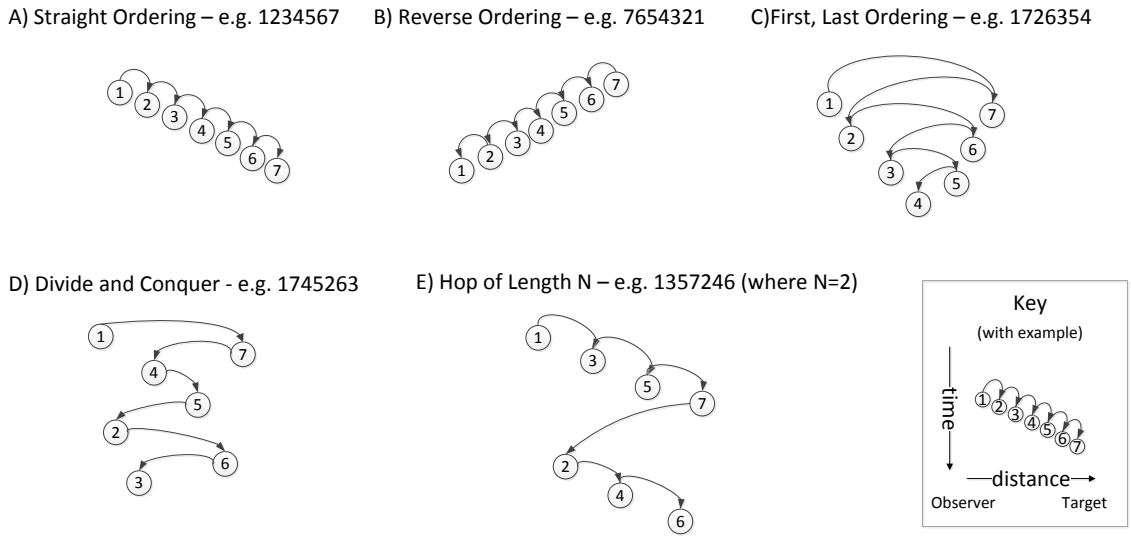



*Figure 3: Example outputs for each of the LoS sample orderings used*

The Python code to generate each sample order is shown below. The sample value is converted into a coordinate by projecting the value as a point along the LoS from the observer towards the target; a scaling factor may be introduced to accommodate different raster cell resolutions. The target's visibility is considered on each iteration until there are no more samples remaining (target visible), or the target is considered obscured (early rejection). The code for methods C and D are noticeably longer than the code required to implement the simpler incremental sampling strategies (A, B, E).

```
# [METHOD A] -- STRAIGHT ORDERING

tar_dist=distance(ObstoTar)

for s in range (1,tar_dist+1):
    if (visibility(s)==False):
        return False

return True
```

```
#  [METHOD B] -- REVERSE ORDERING

tar_dist=distance(ObstoTar)

for s in range (tar_dist,0,-1):
    if (visibility(s)==False):
        return False

return True
```

```
#  [METHOD C] -- FIRST, LAST ORDERING

tar_dist=distance(ObstoTar)

near_marker=0
far_marker=tar_dist+1

for d in range (1,tar_dist+1):
    if (d%2!=0):
        near_marker = near_marker+1
        if (visibility(near_marker)==False):
            return False


    else:
        far_marker = far_marker-1
        if (visibility(far_marker)==False):
            return False

return True
```

```
# [METHOD E] -- HOP ORDERING
tar_dist=distance(ObstoTar)
hop_n=2 #hop size in metres

for offset in range (1,hop_n+1):
    for s in range (offset,tar_dist+1,hop_n):
        if (visibility(s)==False):
        return False

return True
```

```
#  [METHOD D] -- DIVIDE AND CONQUER ORDERING
class Node:
    def __init__(self,l=None,r=None):
        self.left=l
        self.right=r

    def midway(self):
        return (self.right-self.left)/2 + self.left


class Queue (object):
    def __init__(self, q=None):
        if q is None:
            self.q = []
        else:
            self.q = list(q)
    def pop(self):
        return self.q.pop(0)
    def append(self, element):
        self.q.append(element)
    def length (self):
        return len (self.q)

def div_conq (nodelist):
    if (nodelist.length() >0):
        current_node = nodelist.pop()
        midway=current_node.midway()
        if (visibility(midway)==False):
            return False
        if (midway < current_node.right-1):
            n=Node(midway,current_node.right)
            nodelist.append(n)
        if (current_node.left < midway-1):
            n=Node(current_node.left,midway)
            nodelist.append(n)
        return div_conq(nodelist)
    else:
        return True

# ----Main Routine----
tar_dist=distance(ObstoTar)
n=Node(1,tar_dist)
if (visibility(n.left)==False):
        return False
if (visibility(n.right)==False):
        return False

nodelist = Queue()
nodelist.append(n)
div_conq(nodelist)
```

For these trials the application ran on a single thread so that relative differences in observed processing time were the result of changes to the sampling order, rather than any parallelisation implementation. However, this is an embarrassingly parallel case as observer-to-target LoS calculations could be run simultaneously across all available CPU cores, for example sets of agents in an ABM could model visibility to other agents in parallel limited only by the number of processing cores available. Furthermore parallelisation may be included within each LoS so that sets of surface intercept calculations are computed simultaneously; this would incur a minor computational overhead when a thread had determined the target to be obscured making other active threads redundant. However, these parallelisation solutions do not address the issue central to this research – namely performance improvements from different sampling order strategies and the reduction of processing overhead (e.g. to minimise battery usage).

Trials were conducted using a PC with 3GB RAM and a 2.4GHz Intel Duo CPU with power management set to high performance mode, recording the processor execution time to avoid timing variations resulting from other OS background processes. The software was written in C# .NET 4.5.1 using Visual Studio 2013 with code optimisation enabled. The DSM was loaded into memory at the start of the experiment to remove variations from disk activity. These initial trials were conducted using a 6.7km by 4.3km DSM at 1m resolution for the city of Edinburgh, Scotland (Figure 4).

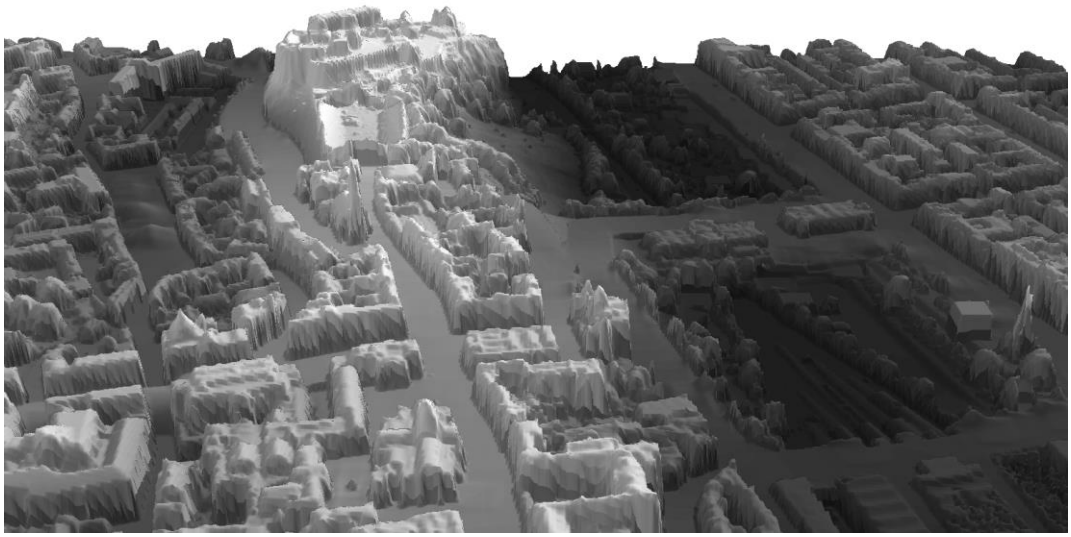


*Figure 4: Perspective view of a section of Digital Surface Model used for trials in Edinburgh (Scotland), looking towards Edinburgh Castle with Old Town on the left, and New Town on the right*

### 4.1 Trial 1 – Single Point to Point

Random pairs of observer-target locations were selected within the study region (Edinburgh) until a pair were found where the target was calculated to be visible using the golden case (i.e. scanning all points along the LoS) and the observer was located in a pedestrian accessible space (i.e. road, open space), to simulate observers at ground level and avoid calculations from roof tops. The selected observer-target pair was a distance of 1.2km apart, but to increase the workload for timing purposes the visibility calculation was carried out 5000 times in succession without caching.

The results indicated no performance benefit among the approaches A,B,C and E, as all intermediate cells are sampled between the observer and target (Table 1 – Visible case). Minor performance differences are due to overheads in the ordering algorithm implementations. Method D showed an increased calculation time (121%) due to the more complex queueing and dequeueing methods required to generate the sample order, as seen in the code outlined previously.

A second observer-target pair was then randomly chosen with the conditions of also being 1.2km apart and pedestrian accessible, but where the target was not visible. The CPU times for those calculations are shown in Table 1 (Not Visible case), and reveal significantly shorter calculation times than the visible case. These faster times result from early rejections made possible when a target is calculated to be obscured, and allowing the LoS to terminate before scanning all intermediate cells. The quicker execution times are evident for all methods, but the most significant performance gains arise from the methods that distribute the sampling locations along the ray length rather than scan incrementally (i.e. methods A,B,C). Notably Method D (divide and conquer) returned 'target obscured' in 13.2% of the time taken by Method A (straight ordering), while Method E (hop) completed the task in 27.2% of the time of Method A.

The performance benefits from the not visible case will be influenced by the specific topography and observer-target locations used in the trial but this initial trial does reveal differences in the overheads for visible cases, and that sample ordering does impact overall LoS efficiency. Further trials were undertaken with a random mix of visible and non-visible targets to establish the typical benefits from LoS sample reordering in urban landscapes.

*Table 1: Visibility trials using different sampling orders for a visible and non-visible observer-target pairs 1.2km apart. CPU times are given in seconds for 5000 trials, the hop distance (Method E) was 5m.*

| Case | Order | A | B | C | D | E (N=5m) |
|---|---|---|---|---|---|---|
| Visible | Time (sec) | 3.837 | 3.900 | 3.884 | 4.648 | 3.900 |
| | % of A | 100.0 | 101.5 | 101.0 | 121.1 | 101.5 |
| Not Visible | Time (sec) | 0.114 | 0.202 | 0.233 | 0.015 | 0.031 |
| | % of A | 100.0 | 177.2 | 204.4 | 13.2 | 27.2 |

## 4.2 Trial 2 – Multiple Observer-Target Pairs

For the second set of trials, locations were selected randomly around Edinburgh city with a restriction that they must be in pedestrian accessible locations, as shown in Figure 5. Edinburgh is a hilly city, and consists of an Old Town with narrow winding streets (to the south), and a New Town (to the north) which has wider roads and a more grid like street pattern. Trial 2A was conducted using 2000 locations within the Old Town, while Trial 2B used 2000 locations within the New Town. The trials involved casting a LoS from each sample location to all others in that trial set, resulting in 4 million ray casts per trial. An additional Trial 2C was conducted using the combined 4000 locations to include views between Old and New Towns, resulting in 16 million rays cast.

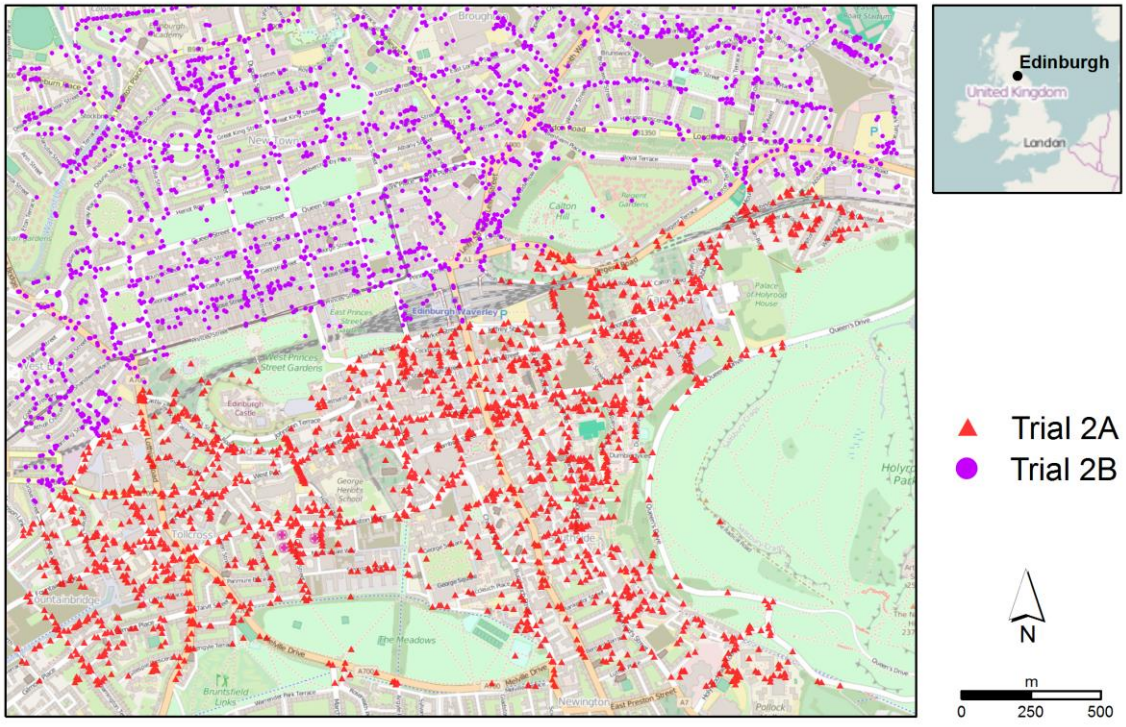



*Figure 5: Randomly selected locations in Edinburgh (Scotland) for Trial 2 (Background map: © OpenStreetMap contributors)*

A check was carried out after each trial to ensure the results matched the golden case (Method A). The calculations were not reflexive as an elevation offset of 1.8 metres was applied to the observer, to model the user's eye-height, and an elevation offset of 0.5 metres was applied to the target to reduce impact of minor data noise in the surface model. The results from these trials are shown in Table 2, with times given in seconds, and where the hop size for Method E was set to 5 metres.

*Table 2: Comparing the performance differences in sample order strategies for computing the visibility from locations around Edinburgh's Old town (2A) and New town (2B) and the combined samples from both regions (2C). CPU times are given in seconds, and the hop distance for E was 5m.*

| Trial | Sample Order | A | B | C | D | E (N=5m) |
|---|---|---|---|---|---|---|
| 2A | Time (sec) | 89.653 | 89.295 | 74.880 | 17.893 | 28.048 |
| | % of A | 100.0 | 99.6 | 83.5 | 19.9 | 31.3 |
| 2B | Time (sec) | 86.892 | 99.232 | 80.683 | 16.348 | 30.076 |
| | % of A | 100.0 | 114.2 | 92.8 | 18.8 | 34.6 |
| 2C | Time (sec) | 339.212 | 334.426 | 293.668 | 62.731 | 103.907 |
| | % of A | 100.0 | 98.6 | 86.6 | 18.5 | 30.6 |

As a percentage the calculation times are fairly consistent across the different trials, despite different street patterns and topography, ranking D, E, C, A/B in order of performance from most to least efficient.

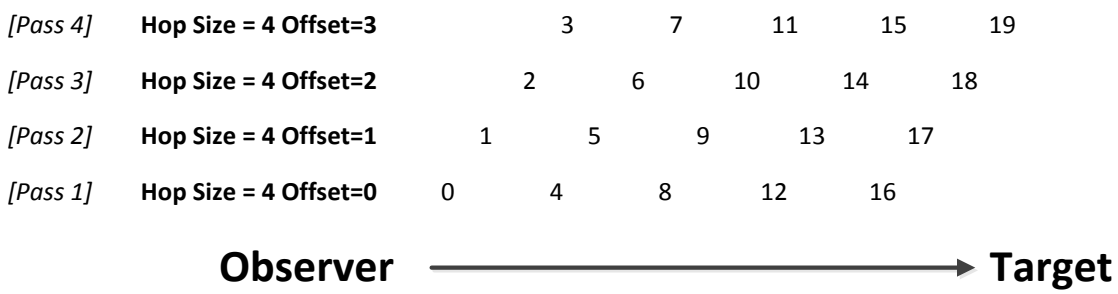
This more comprehensive trial showed the additional computational overhead from sample reordering was outweighed in the majority of cases (i.e. all cases apart from 2B-B), and confirmed that the most impressive reductions resulted from strategies that spread the sample locations along the LoS – the divide and conquer approach (D), and the hop method (E).

### 4.3 Trial 3 – Varying the hop size

In the previous example the hop size was set to 5 metres, however to determine if a more appropriate value could be used, two further trials were conducted using the same

411 random locations from Trials 2A and 2B but adjusting the hop size value on each run.
412 As before each trial resulted in 4 million LoS to calculate the intervisibility between the
413 2000 sample locations. The locations for these trials were centred on different parts of
414 the city in order to assess the sensitivity of the hop value to urban morphology.

415 For larger hop sizes fewer samples are required on each pass but there is an
416 increase in the number of subsequent 'in filling' passes to ensure that all the sample
417 locations are sampled (Figure 6). This is necessary so that all cells are sampled along
418 the LoS from observer to target in visible cases. The optimum hop size occurs when
419 there is the highest chance of an early rejection of the target's visibility, when the tested
420 viewing angle is greater than that of the target.
421




422

423 *Figure 6: Example of the hop sample ordering where hop size is 4 and the offset is*
424 *incremented by 1*
425
426
427 The results from these two trials are shown in Figure 7, and exhibit very similar
428 patterns, whereby the optimum hop size is between 25-35 metres. The optimal hop size
429 for Trial 2A was 25m, while hops of 30m yielded marginally faster times in Trial 2B.
430 The greatest performance gains are made until the hop size reaches 15m, after which
431 there is a plateau until around 40 metres. Beyond 40m there is a marginal increase in
432 execution time, as more infilling sample locations are required. In cases where the hop
433 size is greater than the distance from observer-target then only the offset parameter
434 plays a part, and effectively the sampling strategy mimics Method A (straight ordering).
435

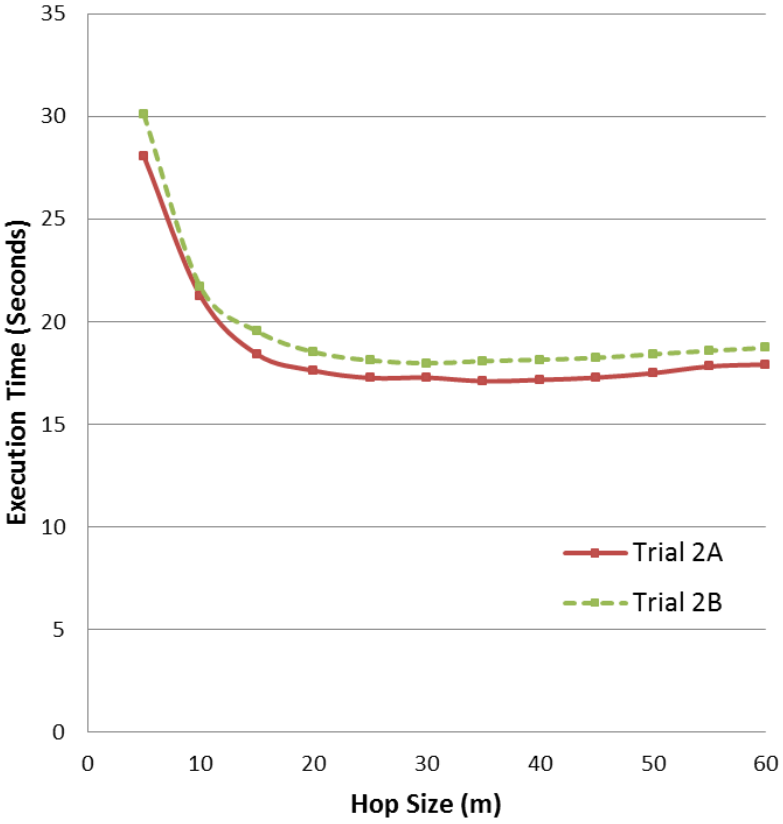



*Figure 7: Execution times with varying hop sizes for two trials in different parts of Edinburgh City (Trial 2A - Old Town and Trial 2B - New Town)*

Table 3 summarises the findings for the improved sampling strategies D and E, compared to the incremental forward scan (A). After adjusting the hop sized this offered comparable speed improvements to the Divide and Conquer method, yet without any overheads in visible cases (as per Table 1).

*Table 3: Result Comparison Summary – Edinburgh trials*

| Trial | Order | A | D | E | E |
|---|---|---|---|---|---|
| 2A | Time (seconds) | 89.653 | 17.893 | 28.048 (N=5m) | 17.253 (N=25m) |
| | % of A | 100 | 19.9 | 31.3 | 19.2 |
| 2B | Time (seconds) | 86.892 | 16.348 | 30.076 (N=5m) | 17.971 (N=30m) |
| | % of A | 100 | 18.8 | 34.6 | 20.7 |

## 4.4 Trials for Other Cities

Further trials were undertaken in two other UK cities (Birmingham and Nottingham) to determine if methods D and E exhibited similar performance benefits in different urban morphologies. These cities were chosen due to their different street patterns, topography, and the availability of LiDAR datasets from which 1 metre resolution DSMs could be generated. The trials were conducted using a 4km by 4km region around the city centre, and as before 2000 random pedestrian accessible locations were selected around the city (Figure 8). As with the Edinburgh trials particular attention was taken to ensure that sample locations did not fall on building or tree top locations. This was done by rejecting randomly selected sample locations where focal statistics indicated nearby road elevation values were more than 1m lower than the selected cell (i.e. current road cell is 1 metre or more above the surrounding road level).

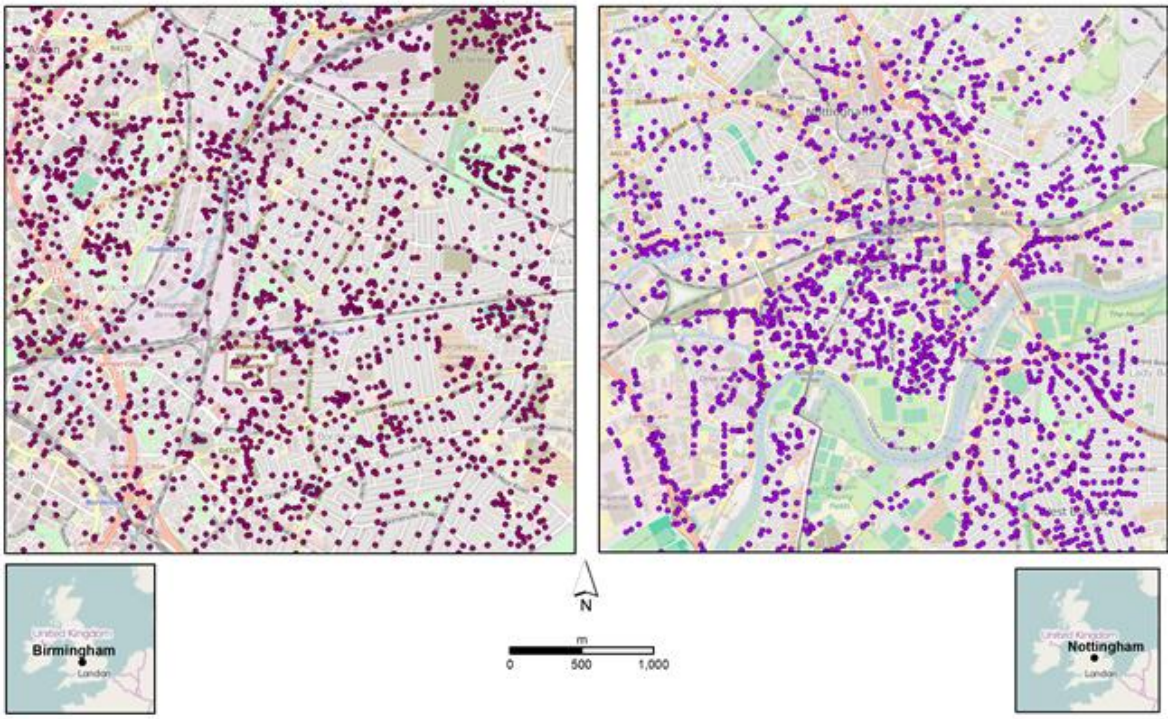



*Figure 8: Sample locations for Birmingham and Nottingham city trials (Background map: © OpenStreetMap contributors)*

The intervisibility from each sample to all others within each city was calculated using Methods A-E (Table 4). The results exhibit a very similar pattern to that noted for the Edinburgh trials (Table 2).

*Table 4: Visibility calculations times for each sampling methods in Nottingham and Birmingham cities*

| Trial | Sample Order | A | B | C | D | E (N=5m) |
|---|---|---|---|---|---|---|
| Birmingham | Time (sec) | 98.655 | 122.257 | 80.808 | 18.142 | 31.153 |
| | % of A | 100.0 | 123.9 | 81.9 | 18.4 | 31.6 |
| Nottingham | Time (sec) | 61.916 | 75.925 | 49.951 | 16.536 | 23.524 |
| | % of A | 100.0 | 122.6 | 80.7 | 26.7 | 37.9 |

As before method D shows the biggest performance improvement completing the calculations in 18.4% (Birmingham) and 26.7% (Nottingham) of method A. The optimum hop size was 30 metres for Nottingham, and 25 metres for Birmingham (Figure 9), though it should be noted that beyond 20m the impact on execution time is minimal and for this range of urban morphologies individualised hop sizes are not necessary.

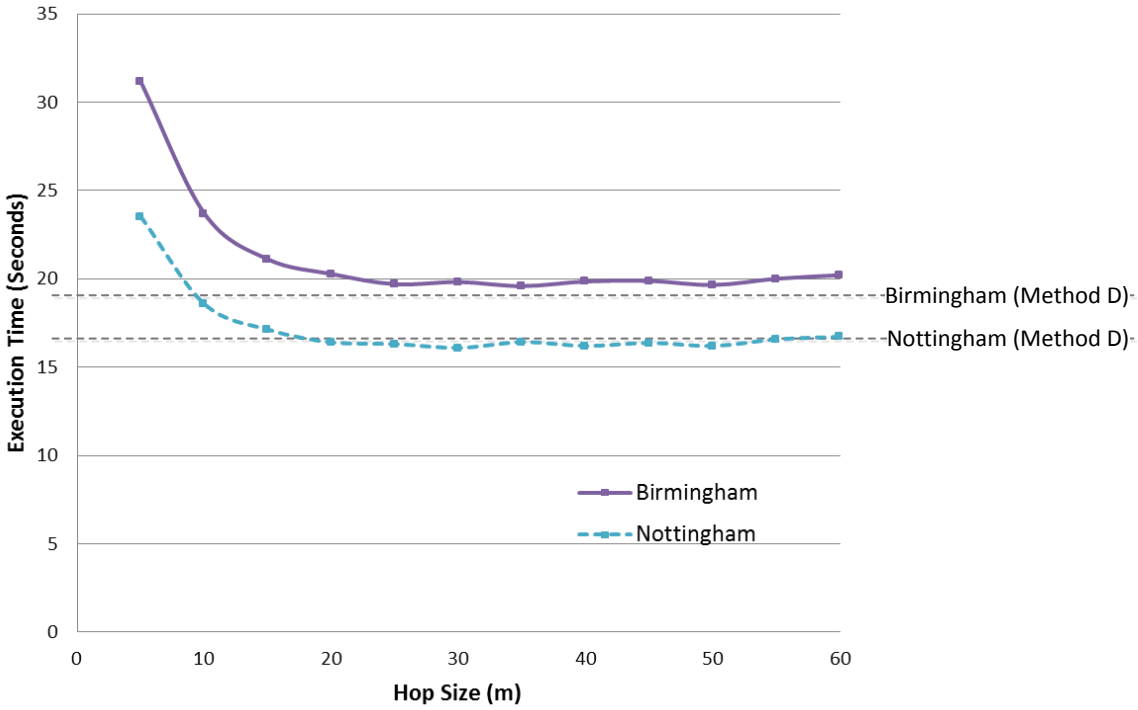



*Figure 9: Trials to determine optimum hop size for Birmingham and Nottingham cities, showing Divide and Conquer times (Method D) for each city as a comparison*

These results for Birmingham and Nottingham show a similar trend to that noted in the Edinburgh trials, with processing times reducing significantly until the hop size reaches 15 metres. From 20 metres upwards the performance benefits are fairly consistent, with a minor decrease in performance as the hop size increases above 50 metres.

Based on the trials across three UK cities, of varying urban morphology, the results demonstrate a consistent performance improvement in using a Divide and Conquer approach, or a hopping strategy with a hop size of between 25 metres and 30 metres.

The DSM query rate can be calculated for each method as a ratio of the number of DSM queries required when casting a ray from observer to target divided by the processing time. Table 5 shows the average number of samples per millisecond achieved by each method, across all of the city trials. It reveals that despite the shorter processing times of the Divide and Conquer approach (D) it has the slowest sampling rate, due to the added complexity of the algorithm, meaning each iteration took longer. However as the number of iterations required was lower than other approaches this method would be the most suitable where DSM access had a high cost, such as for very large terrains stored on disk rather than in memory.

*Table 5: Efficiency of method based on DSM query rate*

| **Method** | **A** | **B** | **C** | **D** | **E** |
|---|---|---|---|---|---|
| **Average Samples per Millisecond** | 1398.4 | 1092.4 | 1344.7 | 689.6 | 1186.1 |

Based on the sample rates achieved it is possible to illustrate the benefit of using a point-to-point sampling strategy rather than generating a viewshed for each observer in multiple-observer scenarios where Boolean target visibility results are sufficient. At a rate of 1398 samples per millisecond (Method A) it would take at least 11 seconds to compute a viewshed for the 4km by 4km DSM at 1m resolution. As a comparison the same PC takes around 23 seconds to compute a viewshed using ESRI ArcGIS 10.2 for a single observer on the Birmingham DSM. Therefore to calculate a viewshed for each of the 2000 observers, to establish which targets are visible, would take in excess of 6

hours. In comparison the Boolean visibility of observer-target pairs using Ordering Methods D or E is around 30 seconds or better (see Table 3 and Table 4). This illustrates that despite possible repetitions of calculations for intermediate DSM cells there is still a big advantage in using LoS rather than viewsheds in such applications.

To gain a better understanding of the relationship between urban morphology and hop size a further trial was undertaken using a synthetic city design, as explained in the next section.

### 4.5 Analysis of Results and Synthetic City Trials for the Hop Method (E)

It is likely that the geographical scale of urban space has an impact on the hop size, as despite differences in the average building sizes in the three cities (Birmingham 95.1 sq m; Nottingham 95.7 sq.m; Edinburgh 143.0 sq.m) the road widths are similar at between 15 metres and 30 metres wide. When you consider that the observer locations are most probably near a road, it is likely that a sample will soon encounter a building resulting in an early termination of the LoS. To gain a better understanding of the relationship between hop length and road width, six synthetic cities were constructed. These had a uniform grid structure with buildings of 80 metres by 80 metres. One thousand pedestrian accessible points were chosen randomly, and the LoS from each to all other locations were calculated (1 million LoS calculations per synthetic city). The distance to the first interception with a blocking object (i.e. building) was recorded along with the last interception with that object (Figure 10), giving a range of distances for which a hop length would result in an early termination of the LoS. For example a hop length of between 17m and 75m would result in an early LoS termination in the synthetic city dataset with road widths of 10m (from Table 6).

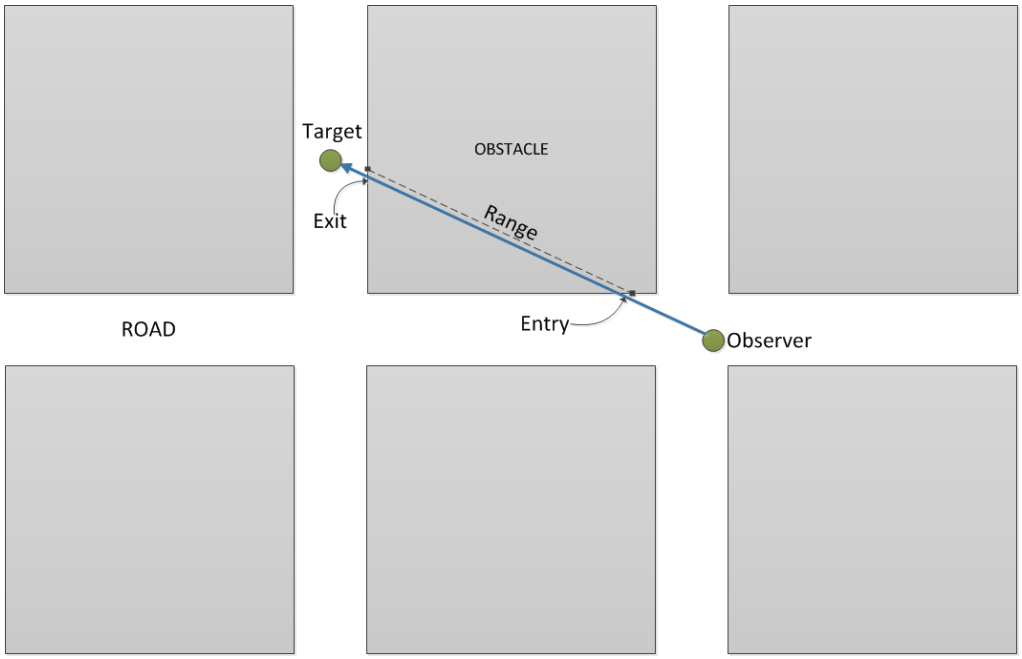



*Figure 10: Illustration of the Entry and Exit Obstacle Distances for LoS in Synthetic City*

*Table 6: Relationship between Road Width and LoS Obstacle Distance for Synthetic Cities for 1 Million LoS calculations (see Figure 11).*

| Road Width (m) | Observer to Obstacle Entry (m) | Observer to Obstacle Exit (m) | Range (m) |
|---|---|---|---|
| 10 | 17 | 75 | 58 |
| 15 | 23 | 83 | 60 |
| 20 | 32 | 92 | 60 |
| 25 | 40 | 100 | 60 |
| 30 | 48 | 109 | 61 |
| 35 | 56 | 117 | 61 |

The Pearson product moment correlation between road width and the first obstacle distance is 0.999 (3dps), showing a very strong positive relationship that effective hop

size increases with wider roads. In these trials on synthetic cities the building size was fixed at 80 metres by 80 metres in all cases, and it is interesting to note that the first intersection was consistently around 1.6 times the road width, and the range was a fairly consistent 60 metres in these cases with the 80 metre square buildings.

Analysis of the trial data exhibits similar patterns but with the added complexities of a more diverse urban fabric and topography. Figure 11 shows the geographical pattern of performance gains that can be made using a dispersed sample strategy (here a comparison is made to Method E but the results are very similar for Method D) as a result of the urban morphology when calculating the visibility to the 2000 target locations in Trial 2B. This section of New Town in Edinburgh was selected as it demonstrated a range of interesting results within a small geographic region.

Narrow streets (e.g. Figure 11, A) have restricted views and require few samples to determine target visibility, thereby limiting the efficiency gains possible. Similarly when the observer is surrounded by tall buildings particular on curved roads (Figure 11, C) the opportunity for sample reduction is minimal. The largest efficiency gains are made in the more open expanses such as North Bridge (Figure 11, B), and where wide entry and exit roads meet at a roundabout (Figure 11, D), on the longer wider straight roads (Figure 11, E) and at junctions (Figure 11, F) where the space is more open giving way to longer views. In these places the dispersed sampling strategies reduce the total number of samples required dramatically (Method E up to 17 times reduction, while Method D up to 47 times reduction for Trial 2B). The correlation in efficiency gains between Method D and Method E (hop = 30m) when compared to Method A was 0.872 (3dps), indicating that both methods benefit similarly from the surrounding geography.

The overall performance of the various algorithms is a result of the reduction in the number of samples required based on the geographic surroundings, and also the time taken per sample based on the ordering efficiency. In narrow corridors the room for performance improvement is minimal, while greater efficiencies can be made in the more open spaces, along wider streets, and near junctions.

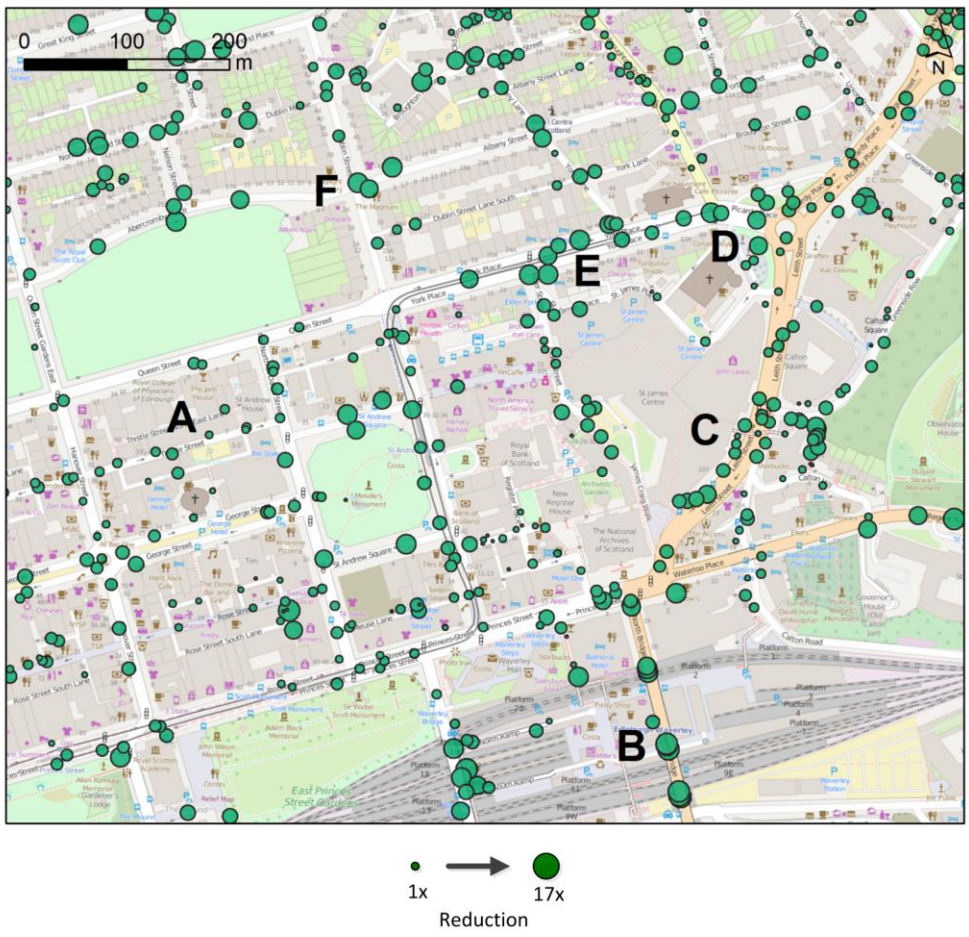



*Figure 11: An excerpt from the map of New Town, Edinburgh, showing the reduction for Trial 2B in total samples used by Method E (N=30m) compared to Method A (Background Map : © OpenStreetMap contributors)*

# 5 Conclusion

This research has shown that sampling order has a significant impact on the calculation times for computing Boolean visibility for observer-target pairs. The two most impressive performance improvements resulted from methods which distribute the samples from observer to target, rather than scan incrementally. A Divide and Conquer approach, which recursively subdivided the space, offered a robust fivefold increase in

599 execution performance, however the more complex algorithm overheads led to an
600 increase in processing times for visible cases. A hopping strategy also showed a fivefold
601 performance increase when using a sampling hop of around 30 metres, with minimal
602 processing overhead for visible cases. Trials in three UK cities revealed that the
603 improvement was robust for hop lengths in the range 20-40 metres. Future work could
604 compare the results across a wider range of cities in different parts of the world, and
605 also in more rural areas to determine the range of hop sizes suited to different
606 topographies. Trials on synthetic datasets indicate the hop length and road width are
607 strongly related. This fact considerably simplifies the tuning of the hop algorithm.

608 This work is not beneficial for searching for site locations where all cells in a
609 study region need to be calculated to determine suitable candidates. In these cases a
610 sweep algorithm would be more suitable (Franklin and Ray 1994, Andrade *et al.* 2011).
611 However the optimisation presented in this paper is suitable where a Boolean visibility
612 result is required in point-to-point scenarios, particularly in multi-observer dynamic
613 situations. This algorithm has been used successfully implemented in a client-server
614 setup to support natural language generation of wayfinding instructions for LBS clients,
615 and is soon to be included in an urban simulation package. Other examples of where it
616 could be used include person-to-person evasion (e.g. military applications), geosensor
617 networks (e.g. determining if other sensors are in direct line of sight), event triggering
618 (e.g. building entrance or junction visibility in LBS applications), location based
619 gaming, and simulations (e.g. crowd modelling, predator-prey agent based models).
620 Salomon *et al* (2004) suggested that up to 40% of processing time can be attributed to
621 LoS queries in simulations, and there is an increasing need to optimise as advances in
622 acquisition and storage technologies have enabled ever greater precision in reality
623 modelling with a commensurate increase in the number of obstacles against which LoS
624 queries must be tested.

625 There are also benefits for client-server setups where a server may be supporting
626 many concurrent users (e.g. friends in view on a location based service), or where
627 calculations are carried out on mobile devices with more limited processing and power

628 resources (e.g. smartphone clients). In these cases reducing the actual computation
629 (rather than increasing available processing power through parallelisation) significantly
630 reduces power consumption.

631 This work has assessed the relative efficiency of LoS algorithms for 'visible /
632 not visible' cases in the context of urban environments. The work has wide ranging
633 importance particularly in the context of highly dynamic urban environments where
634 mobile devices are required to process very large volumes of data, and agent based
635 modelling. It is also relevant to applications that span gaming (in real and synthetic
636 worlds), location based services, and augmented reality more generally. Future work
637 will look at its suitability in rural areas, where the topography is less angular and more
638 softly undulating.

639 # 6 Acknowledgements

640 The research leading to these results has received funding from the EC's 7th
641 Framework Programme (FP7/2011-2014) under grant agreement no. 270019
642 (SpaceBook project). The authors would also like to thank Dr Elijah Van Houten for
643 initial discussions on sampling strategies.

644

645 # 7 References

646 Andrade, M. V. A., Magalhaes, S. V. G., Magalhaes, M. A., Franklin, W. R. & Cutler, B.
647 M., 2011. Efficient viewshed computation on terrain in external memory.
648 *GeoInformatica,* 15**,** 381-397.
649 Baer, W., Baer, N., Powell, W. & Zografos, J., 2005. Advances in terrain augmented
650 geometric pairing algorithms for operational test. *ITEA Modelling and Simulation*
651 *Workshop.* Las Cruces, NM.
652 Bartie, P. & Kumler, M. P., 2010. Route ahead visibility mapping: A method to model
653 how far ahead a motorist may view a designated route. *Journal of Maps,* April
654 2010**,** 84-95.

655 Benedikt, M. L., 1979. To take hold of space: isovists and isovist fields. *Environment and*
656 *Planning B,* 6**,** 47-65.
657 Carver, S. & Washtel, J., 2012. Real-time visibility analysis and rapid viewshed
658 calculation using a voxel-based modelling approach. *GISRUK.* April 6-9,
659 Lancaster University.
660 Davidson, D. A., Watson, A. I. & Selman, P. H., 1993. An evaluation of GIS as an aid to
661 the planning of proposed developments in rural areas. *In* Mather, P. M. (ed.)
662 *Geographical Information Handling: Research and Applications.* London, Wiley.
663 De Floriani, L. & Magillo, P., 1994. Visibility algorithms on triangulated digital terrain
664 models. *International Journal of Geographical Information Systems,* 8**,** 13-41.
665 De Floriani, L., Magillo, P. & Puppo, E., 2000. VARIANT: A system for Terrain
666 modeling at variable resolution. *GeoInformatica,* 4**,** 287-315.
667 De Floriani, L., Marzano, P. & Puppo, E., 1994a. Line-of-sight communication on
668 terrain models. *International Journal of Geographical Information Systems,* 8**,** 329-
669 342.
670 De Floriani, L., Montani, C. & Scopigno, R., 1994b. Parallelizing visibility computations
671 on triangulated terrains. *International Journal of Geographical Information Systems,*
672 8**,** 515-531.
673 De Smith, M. J., Goodchild, M. F. & Longley, P., 2007. *Geospatial Analysis: A*
674 *Comprehensive Guide to Principles, Techniques and Software Tools*: Troubador
675 Publishing.
676 Fisher-Gewirtzman, D. & Wagner, I. A., 2003. Spatial openness as a practical metric for
677 evaluating built-up environments. *Environment and Planning B: Planning and*
678 *Design,* 30**,** 37-49.
679 Fisher, P. F., 1993. Algorithm and implementation uncertainty in viewshed analysis.
680 *International Journal of Geographical Information Science,* 7**,** 331-347.
681 Fisher, P. F., 1996. Extending the applicability of viewsheds in landscape planning.
682 *Photogrammetric Engineering and Remote Sensing,* 62**,** 1297-1302.
683 Franklin, W. R. & Ray, C. K., 1994. Higher isn't necessarily better: Visibility algorithms
684 and experiments. *In* Waugh, T. C. & Healey, R. G. (eds.) *Advances in GIS*
685 *Research: Sixth International Symposium on Spatial Data Handling.* London, Taylor
686 and Francis.
687 Gal, O. & Doytsher, Y., 2012. Fast and Accurate Visibility Computation in a 3D Urban
688 Environment. *GEOProcessing 2012, The Fourth International Conference on*
689 *Advanced Geographic Information Systems, Applications, and Services.* 105-110.
690 Gao, Y., Yu, H., Liu, Y., Liu, Y., Liu, M. & Zhao, Y., 2011. Optimization for viewshed
691 analysis on GPU. *Geoinformatics, 2011 19th International Conference on.* IEEE, 1-5.
692 Groves, P. D., 2011. Shadow matching: A new GNSS positioning technique for urban
693 canyons. *Journal of Navigation,* 64**,** 417-430.

694 Kumler, M. P., 1994. An intensive comparison of Triangulated Irregular Networks
695 (TINs) and Digital Elevation Models (DEMs). *Cartographica: The International*
696 *Journal for Geographic Information and Geovisualization,* 31**,** 1-99.
697 Lee, J., 1991. Comparison of existing methods for building triangular irregular
698 network, models of terrain from grid digital elevation models. *International*
699 *Journal of Geographical Information System,* 5**,** 267-285.
700 Lynch, K., 1976. *Managing the sense of a region*: MIT Press Cambridge.
701 Mills, K., Fox, G. & Heimbach, R., 1992. Implementing an intervisibility analysis model
702 on a parallel computing system. *Computers & Geosciences,* 18**,** 1047-1054.
703 Rana, S., 2003. Fast approximation of visibility dominance using topographic features
704 as targets and the associated uncertainty. *Photogrammetric Engineering & Remote*
705 *Sensing,* 69**,** 881-888.
706 Rana, S. & Morley, J., 2002. *Optimising visibility analyses using topographic features on the*
707 *terrain*. Available from:
708 http://www.casa.ucl.ac.uk/publications/workingPaperDetail.asp?ID=44 [23 June
709 2009].
710 Salomon, B., Govindaraju, N., Sud, A., Gayle, R., Lin, M., Manocha, D., Butler, B.,
711 Bauer, M., Rodriguez, A. & Eifert, L., 2004. Accelerating line of sight
712 computation using graphics processing units. Place: Published, DTIC
713 Document.
714 Seixas, R., Mediano, M. & Gattass, M., 1999. Efficient line-of-sight algorithms for real
715 terrain data. *III SimpÃ³sio de Pesquisa Operacional e IV SimpÃ³sio de LogÃ-stica da*
716 *Marinhaâ€"SPOLM 1999*.
717 Stucky, J. L. D., 1998. On applying viewshed analysis for determining least-cost paths
718 on Digital Elevation Models. *International Journal of Geographical Information*
719 *Science,* 12**,** 891-905.
720 Tandy, C. R. V., 1967. The isovist method of landscape survey. *In* Murray, A. (ed.)
721 *Symposium: Methods of Landscape Analysis.* May 1967, London, England,
722 Landscape Research Group, 9–10.
723 Tarel, J.-P., Charbonnier, P., Goulette, F. & Deschaud, J.-E., 2012. 3d road environment
724 modeling applied to visibility mapping: an experimental comparison.
725 *Proceedings of the 2012 IEEE/ACM 16th International Symposium on Distributed*
726 *Simulation and Real Time Applications.* IEEE Computer Society, 19-26.
727 Turner, A., Doxa, M., O'Sullivan, D. & Penn, A., 2001. From isovists to visibility graphs:
728 A methodology for the analysis of architectural space. *Environment and Planning*
729 *B: Planning and Design,* 28**,** 103-121.
730 Van Kreveld, M., 1996. Variations on sweep algorithms: efficient computation of
731 extended viewsheds and classifications. *In Proc. 7th Int. Symp. on Spatial Data*
732 *Handling.* Citeseer.

733 Wang, J., Robinson, G. J. & White, K., 2000. Generating viewsheds without using
734 sightlines. *Photogrammetric engineering and remote sensing,* 66**,** 87-90.
735 Xia, Y., Li, Y. & Shi, X., 2010. Parallel viewshed analysis on GPU using CUDA.
736 *Computational Science and Optimization (CSO), 2010 Third International Joint*
737 *Conference on.* IEEE, 373-374.
738 Ying, S., Li, L., Mei, Y. & Peng, X., 2006. Incremental terrain visibility analysis.
739 *Proceedings of SPIE - The International Society for Optical Engineering.* 28 October
740 2006, Wuhan, SPIE.
741
742